DSA_MINI_PROJECT

METRO MANAGEMENT SYSTEM

112103150 : Shirshak Tiple
112103152 : Omkar Tupe
112107043 : Vibhav Pande

OBJECTIVE : TO PROVIDE EFFECTIVE PATH FROM ONE STATION TO ANOTHER

Data Structure Used

01 Linked List

02 AVL Tree

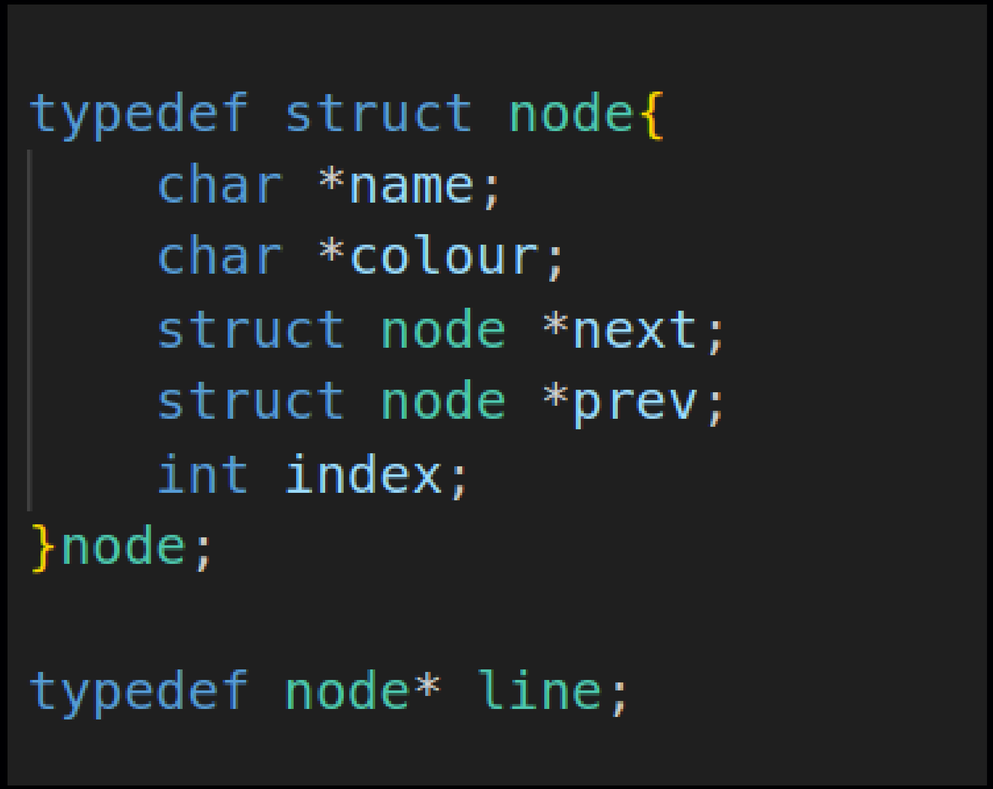
03 Graph (Adjacency List)

04 Queue

LINKED LIST :

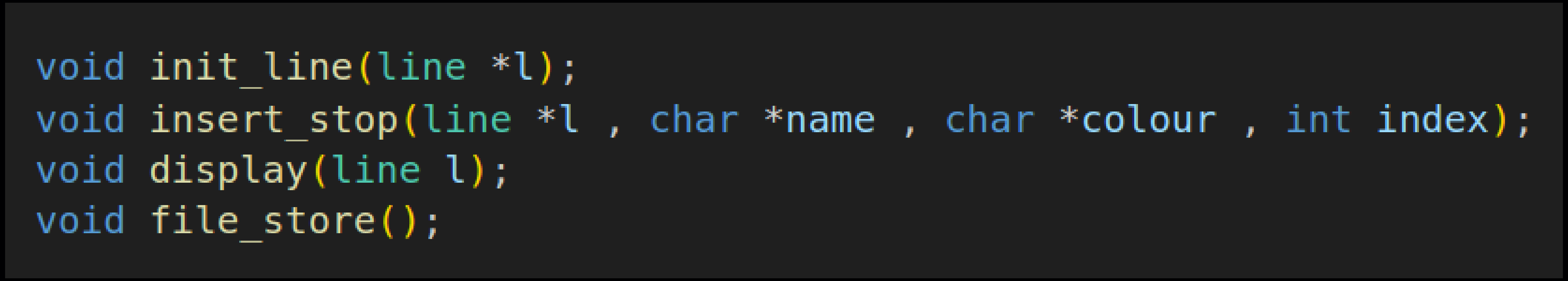
The list used here store the data about the intermediate stations that user needs to travel to reach from source to destination .

Structure used for List :



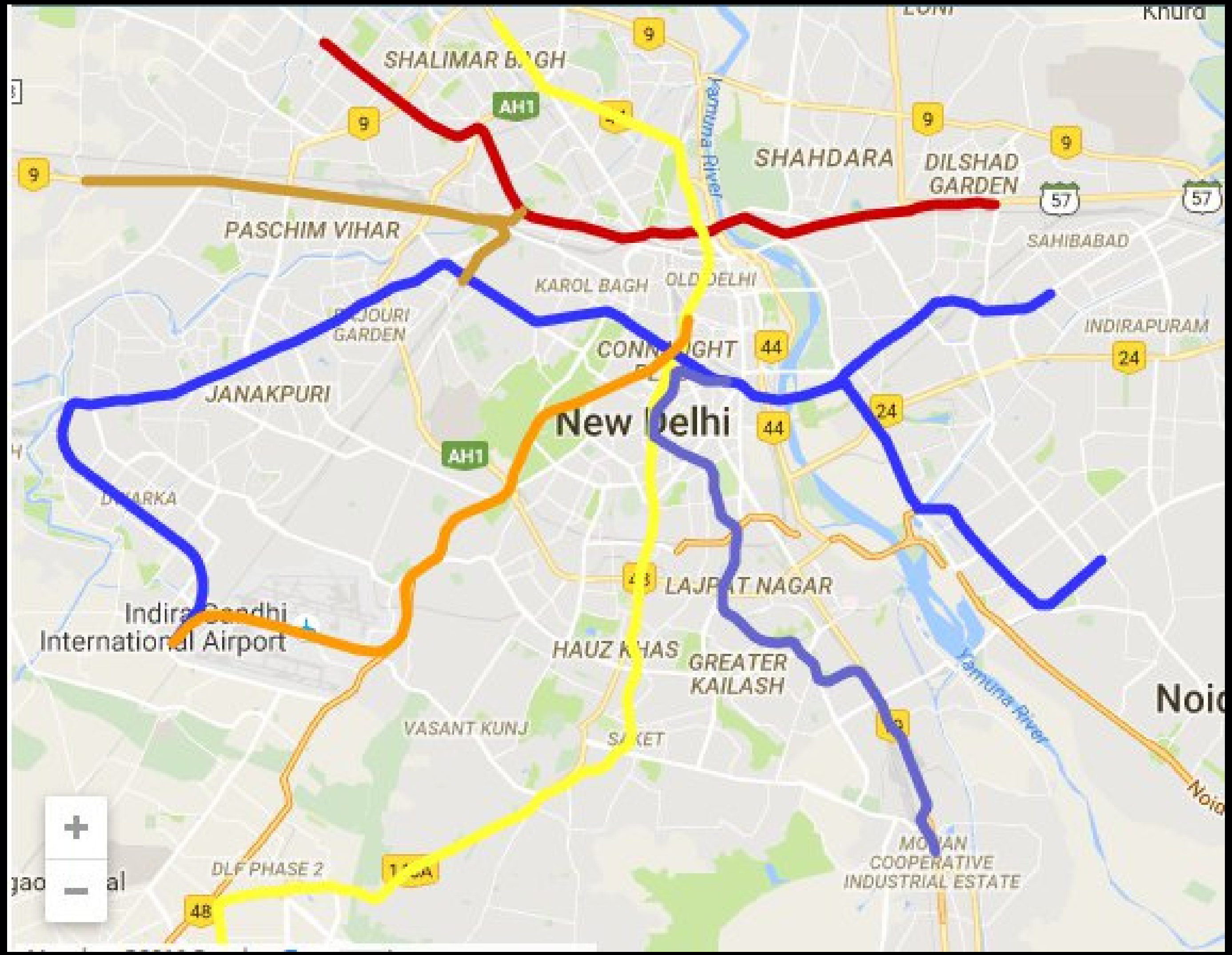
```
typedef struct node{
    char *name;
    char *colour;
    struct node *next;
    struct node *prev;
    int index;
}node;

typedef node* line;
```

Functions of List :



```
void init_line(line *l);
void insert_stop(line *l , char *name , char *colour , int index);
void display(line l);
void file_store();
```

Interconnection of different lines :

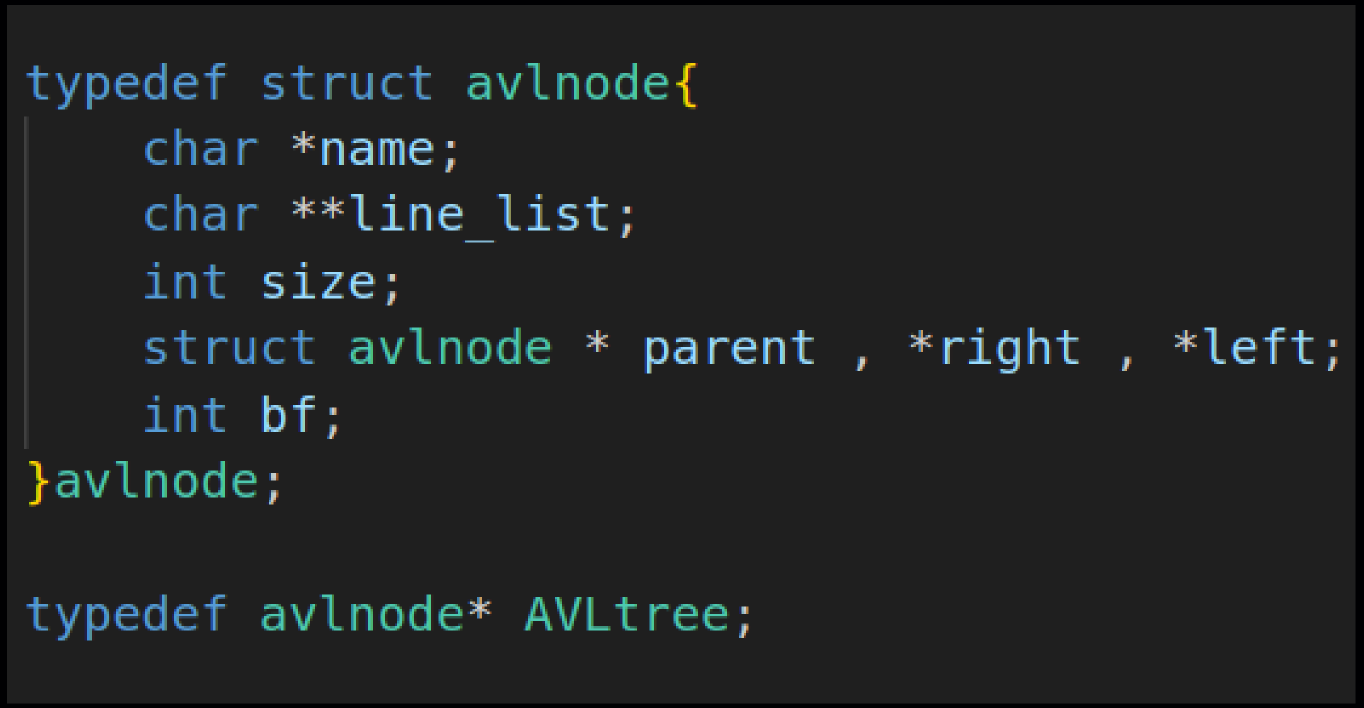


AVL TREE :

AVL Tree which is a balanced tree it is used to store the data about the stations .

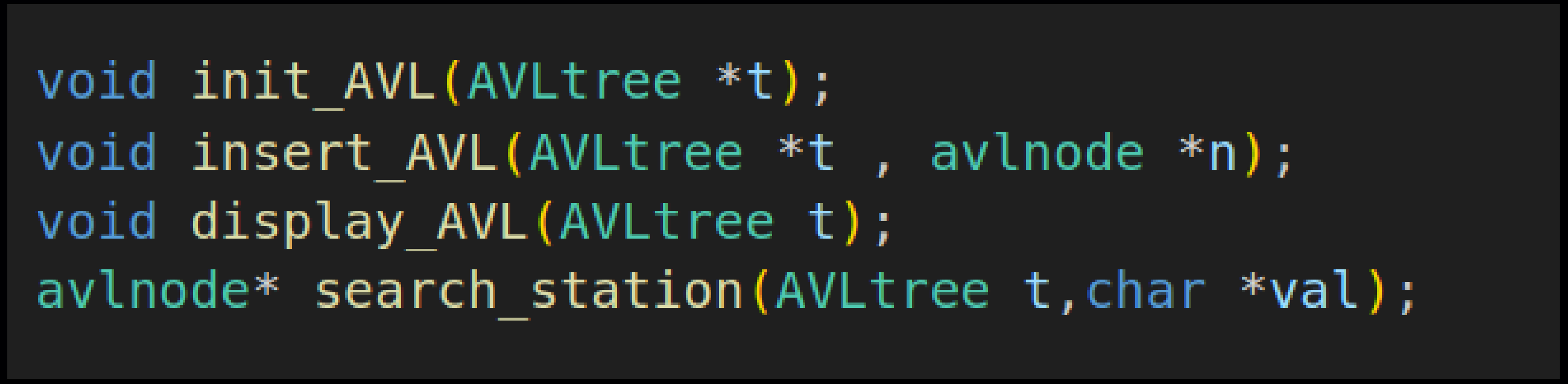
Reason for using AVL Tree is that, for searching it has time complexity O(log(n)) .

Structure used for AVL Tree :



```
typedef struct avlnode{
    char *name;
    char **line_list;
    int size;
    struct avlnode * parent , *right , *left;
    int bf;
}avlnode;

typedef avlnode* AVLtree;
```

Functions of AVL Tree :



```
void init_AVL(AVLtree *t);
void insert_AVL(AVLtree *t , avlnode *n);
void display_AVL(AVLtree t);
avlnode* search_station(AVLtree t,char *val);
```

Representation of AVL Tree :

```
             Rajiv Chowk (Yellow)
            /                    \
    New Delhi (Yellow)        Chandni Chowk (Yellow)
           /                   /                     \
Karol Bagh (Blue)     Kashmere Gate (Red)       HUDA City Centre (Yellow)
                         /         \
              Rithala (Red)   Dilshad Garden (Red)
```

DSA_MINI_PROJ

GRAPH :

Graph is used to find the common stations when switching from one line to another line .

By using BFS or DFS we get the list of stations where the two lines intersect .

Line : Here line refers to different colour lines . Ex : Red Line,Violet Line and so...

QUEUE :

Queue works on the principle of FIFO .

We have used queue to perform BFS which gives us the list of intersecting stations .

Line : Here line refers to different colour lines . Ex : Red Line,Violet Line and so...

Algorithm :

Step 1 : Get the source and destination station name from user . Search in the AVL Tree to get the nodes for the station

Step 2 : Then use a function to get on which colour line the station lies .

Step 3 : Then source and destination lies on same line just search for the route from source to destination on that line and return the list of stations

Step 4 : If they lie on different colour lines , use graph to find the list of intersecting stations in between those two colour lines

Store the output as : Source --> Intersecting Station -- > Destination

Step 5 : Give the list of stations stored in the list

?

THANK YOU